

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平11-143852

(43) 公開日 平成11年(1999) 5月28日

(51) Int.Cl.<sup>6</sup>

G 0 6 F 15/78

識別記号

5 1 0

F I

G 0 6 F 15/78

5 1 0 G

5 1 0 D

7/58

7/58

A

審査請求 未請求 請求項の数 8 F D (全 10 頁)

(21) 出願番号

特願平9-325433

(22) 出願日

平成9年(1997)11月11日

(71) 出願人 000004226

日本電信電話株式会社

東京都新宿区西新宿三丁目19番2号

(72) 発明者 浦野 正美

東京都新宿区西新宿三丁目19番2号 日本  
電信電話株式会社内

(72) 発明者 深沢 友雄

東京都新宿区西新宿三丁目19番2号 日本  
電信電話株式会社内

(72) 発明者 武谷 健

東京都新宿区西新宿三丁目19番2号 日本  
電信電話株式会社内

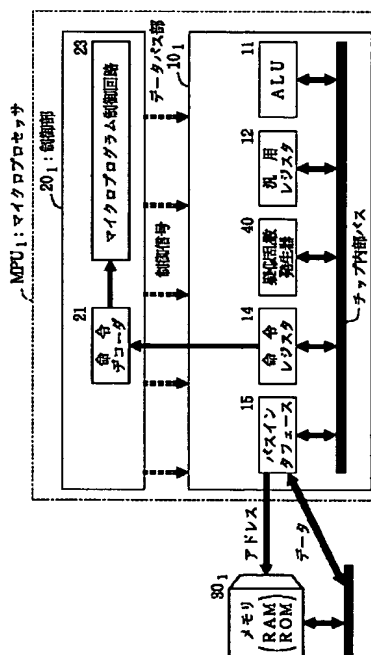
(74) 代理人 弁理士 川久保 新一

(54) 【発明の名称】 マイクロプロセッサ

(57) 【要約】

【課題】 所定のプログラムが格納されているメモリから、その所定のプログラムが読み取られたとしても、その読み取られたプログラムを解析することが困難であるマイクロプロセッサ等を提供することを目的とするものである。

【解決手段】 1つずつ増加する数列以外の数列を非1インクリメント数列と呼んだ場合、この非1インクリメント数列を発生する疑似乱数発生手段を設け、この疑似乱数発生手段が発生する値をアドレスとし、メモリからプログラムを読み出すプログラム読出し手段を設けたものである。



BEST AVAILABLE COPY

## 【特許請求の範囲】

【請求項1】 1つずつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段と；上記疑似乱数発生手段が発生する値をアドレスとし、メモリからプログラムを読み出すプログラム読出し手段と；を有することを特徴とするマイクロプロセッサ。

【請求項2】 請求項1において、上記疑似乱数発生手段は、その初期値を任意に設定可能な手段であることを特徴とするマイクロプロセッサ。

【請求項3】 請求項1において、上記疑似乱数発生手段は、LFSRを基本とする手段と、1以外の所定数ずつ増加する数列を出力する手段と、所定数ずつ減少する数列を出力する手段と、所定の法則（1ずつ加算するという法則を除く法則）に従った数を、所定の初期値に加算した数で構成される数列を出力する手段と、所定の法則に従った数を、所定の初期値から減算した数で構成される数列を出力する手段とのうちのいずれか1つの手段であることを特徴とするマイクロプロセッサ。

【請求項4】 1つずつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段によって発生される値をアドレスとし、この発生されるアドレスの順序に従って、プログラムが格納されていることを特徴とするメモリ。

【請求項5】 1つずつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段と；上記疑似乱数発生手段が発生する値をアドレスとし、この発生されるアドレスの順序に従って、プログラムが格納されているメモリと；を有することを特徴とするメモリ装置。

【請求項6】 請求項5において、上記疑似乱数発生手段は、その初期値を任意に設定可能な手段であることを特徴とするメモリ装置。

【請求項7】 請求項5において、上記疑似乱数発生手段は、LFSRを基本とする手段と、1以外の所定数ずつ増加する数列を出力する手段、所定数ずつ減少する数列を出力する手段と、所定の法則（1ずつ加算するという法則を除く法則）に従った数を、所定の初期値に加算した数で構成される数列を出力する手段と、所定の法則に従った数を、所定の初期値から減算した数で構成される数列を出力する手段とのうちのいずれか1つの手段であることを特徴とするメモリ装置。

【請求項8】 所定のメモリと；1つずつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段が発生する値をアドレスとし、この発生されるアドレスの順序に従って、所定のプログラムを格納する格納手段と；を有することを特徴とするメモリ装置。

## 【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、ICカード等に用いられるマイクロプロセッサ等に関するものである。

【0002】

【従来の技術】図9は、従来のマイクロプロセッサMPUの構成例を示す図である。

【0003】従来のマイクロプロセッサMPUは、電子情報通信学会編「ULSI設計技術」の第115頁の図5.2に示されている例であり、通常、データバス部10と、制御部20とによって構成されている。

【0004】データバス部10は、算術論理ユニット（ALU）11と、汎用レジスタ12と、プログラムカウンタ13と、命令レジスタ14と、バスインタフェース15とを有する。制御部20は、命令デコーダ21と、マイクロプログラム制御回路22とを有する。また、マイクロプロセッサMPUは、バスインタフェース15、外部のアドレスバス、データバスを介して、外部のRAM、ROM等のメモリ30と接続されている。

【0005】図10は、上記従来例において、メモリ30に格納されているプログラムの一例を示す図である。

【0006】このメモリ30において、アドレス0000～0111のそれぞれに、プログラムの命令コードop1～op8が格納されている。ここでは、アドレスとして0111までを例示してあるが、それ以後の番地にも、上記と同様にプログラムの命令コードが格納されている。つまり、所定のアドレスに対応して所定の命令コードが格納され、上記所定のアドレスに1が加算されたアドレスに対応して、上記所定の命令コードの次に実行されるべき命令コードが格納されている。

【0007】従来のマイクロプロセッサMPUにおいて、まず、プログラムカウンタ13に設定された値が、バスインタフェース15を介して外部のアドレスバスに出力されると、アドレスバスに出力されたアドレスの内容を、メモリ30がデータバスに出力する。メモリ30から出力されたデータは、再度、バスインタフェース15を介してマイクロプロセッサMPUの内部バスに送られ、命令レジスタ14にラッチされる。命令レジスタ14にラッチされた命令コードは、制御部20の命令デコーダ21によってデコードされ、マイクロプログラム制御回路22に渡され、マイクロプログラムから各種制御信号が生成され、命令を実行する。

【0008】そして、プログラムカウンタ13に設定されているアドレスに対応する命令が実行されると、通常の命令においては、次に、プログラムカウンタ13に1が加算され、これによって、次のアドレスの内容がメモリ30から読み出され、上記一連の動作が繰り返される。

【0009】図11は、上記従来例におけるプログラムカウンタ13の構成の一例を示す図である。

## 3

【0010】このプログラムカウンタ13は、レジスタ131と、レジスタ131の出力値を1ずつ増加させるインクリメンタ132とによって構成されている。所定のクロックにおいて、レジスタ131に所定の値が設定され、上記所定のクロックの次のクロックで、レジスタ131に設定されている値に1が加算された値をインクリメンタ132が出力し、このインクリメンタ132が出力した値がレジスタ131に設定される。そして、上記所定のクロックから2つ先のクロック以降において、上記一連の動作が順次繰り返される。すなわち、レジスタ131に初期値として0が設定されると、通常の命令では、クロックが発生される度に、設定されている値に1が加算され、この加算された値がレジスタ131に再設定される。

【0011】つまり、上記例において、メモリ30とプログラムカウンタ13とによって、最初に、プログラムカウンタ13に0が設定されるので、0000番地の内容（プログラムの命令コードop1）がメモリ30から読み出され、実行される。次に、プログラムカウンタ13の値に1が加算された値をプログラムカウンタ13が出力し、メモリ30の0001番地に格納されている命令コードop2が読み出され、実行される。以後、プログラムの命令コードop3、op4、op5、op6、op7、op8が順番に読み出され、実行される。それ以後も、上記と同様である。

【0012】図12は、上記従来例におけるマイクロプログラム制御回路22の構成例を示すブロック図である。

【0013】マイクロプログラム制御回路22は、μプログラムカウンタ221と、μプログラムメモリ222と、μ命令デコーダ223とを有するものである。

【0014】命令レジスタ14に設定された内容が、制御部20の命令デコーダ21によってデコードされ、このデコードされた値が、μプログラムカウンタ221に設定される。この設定された値をアドレスとし、μプログラムメモリ222に格納されているμ命令が読み出され、μ命令デコーダ223によってデコードされ、制御信号が生成される。

【0015】次に、プログラムカウンタ13の動作と同様に、μプログラムカウンタ221の内容に1が加算され、この加算された値をアドレスとし、μプログラムメモリ222からμ命令を読み出し、実行する。

【0016】上記のように、従来のマイクロプロセッサMPUでは、プログラムカウンタ13の内容は1ずつ増加する。プログラムカウンタ13が出力した値をアドレスとして、プログラムの内容が順次読み出されるので、プログラムはメモリ30上に順番に格納されている。

【0017】

【発明が解決しようとする課題】ところで、ICカードの応用分野が近年拡大されつつあり、これにつれてIC

## 4

カードのセキュリティが重視されるようになってきている。ICカード内に納められている機密情報を解読しようとする場合、マイクロプロセッサの論理回路、プログラムが容易に解読できる状態にあれば、この情報に基づいて、マイクロプロセッサの内部を解析することができ、これによって、機密情報を解読される可能性がある。

【0018】つまり、従来のマイクロプロセッサMPUのように、プログラムがメモリ30上に順番に格納されていると、メモリ30のメモリセルに格納されているデータを順番に読み取れば、メモリ30に格納されているプログラムを解読することができる。

【0019】本発明は、所定のプログラムが格納されているメモリから、その所定のプログラムが読み取られたとしても、その読み取られたプログラムを解析することが困難であるマイクロプロセッサ、メモリ、メモリ装置を提供することを目的とするものである。

【0020】

【課題を解決するための手段】本発明は、1つずつ増加する数列以外の数列を非1インクリメント数列と呼んだ場合、この非1インクリメント数列を発生する疑似乱数発生手段を設け、この疑似乱数発生手段が発生する値をアドレスとし、メモリからプログラムを読み出すプログラム読出し手段を設けたマイクロプロセッサである。

【0021】

【発明の実施の形態および実施例】図1は、本発明の一実施例であるマイクロプロセッサMPU<sub>1</sub>を示すブロック図である。

【0022】マイクロプロセッサMPU<sub>1</sub>は、図9に示す従来のマイクロプロセッサMPUにおいて、プログラムカウンタ13の代わりに、疑似乱数発生器40を使用するものである。ここで、疑似乱数発生器40は、1つずつ増加する数列以外の数列を「非1インクリメント数列」と呼んだ場合、非1インクリメント数列を発生する疑似乱数発生手段の例である。

【0023】つまり、マイクロプロセッサMPU<sub>1</sub>は、データバス部10<sub>1</sub>と、制御部20<sub>1</sub>とによって構成されている。

【0024】データバス部10<sub>1</sub>は、算術論理ユニット（ALU）11と、汎用レジスタ12と、疑似乱数発生器40と、命令レジスタ14と、バスインタフェース15とを有する。制御部20<sub>1</sub>は、命令デコーダ21と、マイクロプログラム制御回路23とを有する。また、マイクロプロセッサMPU<sub>1</sub>は、バスインタフェース15、外部のアドレスバス、データバスを介して、外部のRAM、ROM等のメモリ30<sub>1</sub>と接続されている。

【0025】図2は、上記実施例における疑似乱数発生器40の構成例と、疑似乱数発生器40が出力する値の例を示す図である。

【0026】疑似乱数発生器40は、図2（1）に示す

10

20

30

40

50

## 5

ように、D-FF401、402、403、404と、Ex. NOR回路405とを有する。D-FF401、402、403、404が直列に接続され、D-FF401、402、403、404が出力する値を $Q_1$ 、 $Q_2$ 、 $Q_3$ 、 $Q_4$ とし、値 $Q_1$ と $Q_4$ とがEx. NOR回路405に入力され、Ex. NOR回路405の出力端子がD-FFのD入力端子に接続されている。また、値 $Q_1$ 、 $Q_2$ 、 $Q_3$ 、 $Q_4$ が疑似乱数発生器40が出力する値（疑似乱数）である。

【0027】次に、疑似乱数発生器40の動作について説明する。

【0028】まず、1つ目のクロックCL1によって、D-FF401、402、403、404の全てに0がセットされていたとする。この場合、Ex. NOR回路405が1を出力する。

【0029】2つ目のクロックCL2によって、Ex. NOR回路405の出力値「1」をD-FF401が出力し、D-FF401、402、403がそれぞれ出力していた値0、0、0と同じ値を、D-FF402、403、404が出力する。つまり、クロックCL2によって、疑似乱数発生器40の出力値（ $Q_1$ 、 $Q_2$ 、 $Q_3$ 、 $Q_4$ ）は、図2（2）に示すように、「1000」になる。この場合、Ex. NOR回路405が「0」を出力する。

【0030】3つ目のクロックCL3によって、Ex. NOR回路405の出力値「0」をD-FF401が出力し、D-FF401、402、403がそれぞれ出力していた値1、0、0と同じ値を、D-FF402、403、404が出力する。つまり、クロックCL3によって、疑似乱数発生器40の出力値（ $Q_1$ 、 $Q_2$ 、 $Q_3$ 、 $Q_4$ ）は、図2（2）に示すように、「0100」になる。この場合、Ex. NOR回路405が「1」を出力する。

【0031】4つ目のクロックCL4によって、Ex. NOR回路405の出力値「1」をD-FF401が出力し、D-FF401、402、403がそれぞれ出力していた値0、1、0と同じ値を、D-FF402、403、404が出力する。つまり、クロックCL4によって、疑似乱数発生器40の出力値（ $Q_1$ 、 $Q_2$ 、 $Q_3$ 、 $Q_4$ ）は、図2（2）に示すように、「1010」になる。この場合、Ex. NOR回路405が「0」を出力する。

【0032】5つ目のクロックCL5以降においても、上記と同様にして、図2（2）に示すように、疑似乱数（ $Q_1$ 、 $Q_2$ 、 $Q_3$ 、 $Q_4$ ）が発生する。

【0033】つまり、疑似乱数発生器40は、常に、所定の乱数を同じ順序で発生するものであり、疑似乱数発生器40によって順次発生される乱数の系列は、再現性のある系列である。

【0034】図3は、上記実施例であるマイクロプロセ

## 6

ッサMPU<sub>1</sub>の動作を説明する図である。

【0035】疑似乱数発生器40は、上記のように、0000、1000、0100、1010、0101、0010、1001、1100、0110、1011、1101、1110、0111、0011、0001の順番で値を発生し、これら発生されるそれぞれの値をアドレスとして使用する。

【0036】そして、プログラムの命令コードop1～op15が、上記アドレスの順番で、メモリ30<sub>1</sub>に内蔵されている。すなわち、最初に発生するアドレス0000番地には命令コードop1が格納され、2番目に発生するアドレス1000番地には命令コードop2が格納され、3番目の値に対応するアドレス0111番地には命令コードop3が格納されている。4番目以降に発生するそれぞれのアドレスには、図3に示すように、上記と同様に、命令コードop4、op5、……が格納されている。

【0037】次に、上記実施例の動作について説明する。

【0038】まず、疑似乱数発生器40に、初期値として「0000」が設定され、この値「0000」をアドレスとし、このアドレス「0000」に対応するメモリ30<sub>1</sub>の内容（命令コード）op1を読み出し、この命令コードop1を実行する。この実行が終わると、疑似乱数発生器40は、クロックCL1に応じて、上記アドレス「0000」の次の値「1000」を発生し、この値「1000」をアドレスとし、このアドレス「1000」に対応するメモリの内容（命令コード）op2がメモリ30<sub>1</sub>から読み出され、命令コードop2を実行する。この実行が終わると、疑似乱数発生器40は、クロックCL2に応じて、上記アドレス「1000」の次の値「0100」を発生し、この値「0100」をアドレスとし、このアドレス「0100」に対応するメモリの内容（命令コード）op3がメモリ30<sub>1</sub>から読み出され、命令コードop3を実行する。

【0039】以下、上記と同様にして、発生された疑似乱数の値をアドレスとし、発生されたアドレスにそれぞれ対応する命令コードop3～op15のそれぞれが順次、読み出され、読み出された命令コードop3～op15のそれぞれを順次、実行する。

【0040】ここで注意すべき点は、疑似乱数発生器40をプログラムカウンタの代わりに使用した場合、その発生された疑似乱数の値をアドレスとして、プログラムをメモリ30<sub>1</sub>に予め格納しておくので、正しい順序でメモリ30<sub>1</sub>からプログラムを読み出し、実行することができる点である。

【0041】ところで、上記実施例において、第三者が、メモリ30<sub>1</sub>の内容を解析しようとし、0000番地から順にメモリ30<sub>1</sub>の内容を読み取ると、このようにして読み取られた命令コードは、op1、op15、

## 7

op6、op14、op3、op5、op9、op13……の順であり、メモリ30<sub>1</sub>上のプログラムを正しく解析することはできず、したがって、その読み出された順で命令コードを実行すると、プロセッサが正常に動作せず、プロセッサの動作を正しく解析することができない。つまり、疑似乱数発生器40が発生する値は、疑似乱数発生器40の回路構成によって決定されるので、疑似乱数発生器40の回路構成を解析しない限り、メモリ30<sub>1</sub>上のプログラムを正しく解析することはできない。

【0042】図4は、疑似乱数発生器40と同様の疑似乱数発生器41の構成例と、疑似乱数発生器41の動作を示すタイミングチャートとを示す図である。

【0043】疑似乱数発生器41は、疑似乱数発生手段の他の例であり、疑似乱数発生器40の代わりに使用することができ、しかも、疑似乱数発生器の初期値を設定可能にしたものである。

【0044】疑似乱数発生器41は、D-FF411、412、413、414と、セクタS1、S2、S3、S4と、Ex. NOR回路415とを有するものである。また、セクタS1、D-FF411、セクタS2、D-FF412、セクタS3、D-FF413、セクタS4、D-FF414が直列に接続され、D-FF411、412、413、414が出力する値をQ<sub>1</sub>、Q<sub>2</sub>、Q<sub>3</sub>、Q<sub>4</sub>とし、値Q<sub>1</sub>とQ<sub>4</sub>とがEx. NOR回路415に入力され、Ex. NOR回路415の出力端子がセクタS1のB入力端子に接続されている。また、セクタS1、S2、S3、S4の各A入力端子に、それぞれ、初期値入力端子I1、I2、I3、I4が接続されている。なお、セクタ信号Sが0であるときに、セクタS1～S4はA入力端子の信号を選択し、セクタ信号Sが1であるときに、セクタS1～S4はB入力端子の信号を選択する。また、値Q<sub>1</sub>、Q<sub>2</sub>、Q<sub>3</sub>、Q<sub>4</sub>が疑似乱数発生器40が出力する値（疑似乱数）である。

【0045】次に、疑似乱数発生器41に初期値設定する動作について説明する。

【0046】疑似乱数発生器41に所望の初期値を設定する場合、初期値入力端子I1～I4のそれぞれに所望の初期値を入力し、セクタS1～S4に供給するセレクト信号を0にし、シフトクロックを入力すると、セクタS1～S4のそれぞれは、初期値入力端子I1～I4における初期値を出力し、これら各初期値がD-FF411、412、413、414のそれぞれに入力される。そして、セレクト信号を1に切り換える。これによって、疑似乱数発生器41における初期値設定動作が終了する。

【0047】セレクト信号を1にすることによって、セクタS1がEx. NOR回路415の出力信号を出力し、セクタS2～S4のそれぞれが、前段のセクタ

## 8

S1～S3の出力信号を出力する。その後は、セレクト信号を1に維持し、クロックを入力する度に、セクタS1がEx. NOR回路415の出力信号を出力し、セクタS2～S4のそれぞれが、前段のセクタS1～S3の出力信号を出力する。

【0048】上記実施例によれば、マイクロプロセッサMPU<sub>1</sub>において、従来使用していたプログラムカウンタ13の代わりに疑似乱数発生器40、41を使用し、その疑似乱数発生器40、41が発生する値をアドレスとしてプログラムをメモリ30<sub>1</sub>に格納してあるので、メモリ30<sub>1</sub>に格納されているプログラムをアドレスの順番に読み取ったとしても、実際の実行順序とは異なり、第三者がプログラムを解析することが困難である。したがって、ICカード等のプロセッサのセキュリティを向上させることができる。

【0049】図5は、本発明の第2の実施例であるマイクロプロセッサMPU<sub>2</sub>を示すブロック図である。

【0050】マイクロプロセッサMPU<sub>2</sub>は、図1に示すマイクロプロセッサMPU<sub>1</sub>において、制御部でマイクロプログラム制御を行う代わりに、ハードワイアードロジックで構成した例である。通常、RISC型のマイクロプロセッサでは、マイクロプログラムを用いず、図5に示す命令デコーダ21におけるハードワイアードロジックによって命令デコードする場合が多い。

【0051】つまり、マイクロプロセッサMPU<sub>2</sub>は、データバス部10<sub>1</sub>と、制御部20<sub>2</sub>とによって構成されている。制御部20<sub>2</sub>は、ハードワイアードロジックを有する命令デコーダ21によって構成されている。

【0052】図6は、本発明の第3の実施例であるマイクロプロセッサMPU<sub>3</sub>を示すブロック図である。

【0053】マイクロプロセッサMPU<sub>3</sub>は、図1に示すマイクロプロセッサMPU<sub>1</sub>において、チップの内部バスをアドレスバスとデータバスとに分離した例である。マイクロプロセッサMPU<sub>3</sub>において、疑似乱数発生器40で生成されたアドレスは、チップ内部のアドレスバスに出力され、バスインタフェースを介して、チップ外部のアドレスバスに出力される。これによって、転送速度が速くなり、制御が容易になる。

【0054】図7は、上記実施例におけるマイクロプログラム制御回路23の具体例を示す図である。

【0055】マイクロプログラム制御回路23は、疑似乱数発生器231と、μプログラムメモリ232と、μ命令デコーダとによって構成され、疑似乱数発生器231として、疑似乱数発生機40と同様のものを使用するものであり、これによって、μプログラムメモリ232に格納されているμプログラムが第三者に解析されることを困難にしたものである。なお、μプログラムメモリ232には、メモリ30<sub>1</sub>と同様に、疑似乱数発生器231が発生する疑似乱数（アドレス）の順にプログラム（命令コード）が格納されている。

【0056】図8は、上記実施例における疑似乱数発生器40を一般化したLFSR (Linear Feedback Shift Register) 42を示す図である。

【0057】LFSR 42は、 $n$ 個のDフリップフロップ $FF_1$ 、 $FF_2$ 、 $FF_3$ 、……、 $FF_n$ と、 $n$ 個の接続点 $c_1$ 、 $c_2$ 、 $c_3$ 、……、 $c_{n-1}$ 、 $c_n$ と、 $n-1$ 個のEx. OR $_1$ 、Ex. OR $_2$ 、Ex. OR $_3$ 、……、Ex. OR $_{n-1}$ とを有する。また、接続点 $c_1$ 、 $c_2$ 、 $c_3$ 、……、 $c_{n-1}$ を実際に接続する場合には、その接続される接続点 $c_1 \sim c_{n-1}$ に対応する出力端子 $Q_1 \sim Q_{n-1}$ を対応のEx. ORの入力端子に接続し、接続点 $c_1$ 、 $c_2$ 、 $c_3$ 、……、 $c_{n-1}$ を実際には接続しない場合には、その接続されない接続点 $c_1 \sim c_{n-1}$ をオープンし、対応するEx. ORを削除し、短絡する。なお、図8中、「=1」は、常に接続するという意味であり、したがって、接続点 $c_n$ は常に接続される。

【0058】つまり、たとえば、接続点 $c_1$ を実際に接続する場合には、その接続される接続点 $c_1$ に対応する出力端子 $Q_1$ を対応のEx. OR $_1$ の入力端子に接続し、たとえば、接続点 $c_2$ を実際には接続しない場合には、その接続されない接続点 $c_2$ をオープンし、対応するEx. OR $_2$ を削除し、前後のEx. OR $_1$ とEx. OR $_3$ とを直接接続する(Ex. OR $_3$ が削除される場合には、Ex. OR $_4$ と直接接続する)。

【0059】このようにして、LFSR 42を、疑似乱数発生器40の代わりに使用することができる。この場合、初期値を設定可能であるようにするには、疑似乱数発生器41のように、各Dフリップフロップ $FF_1$ 、 $FF_2$ 、 $FF_3$ 、……、 $FF_n$ の前段に、それぞれセクタを配置すればよい。なお、図8においては、一般的に書かれるEx. ORで表示してあり、図2に示す疑似乱数発生器40、図4に示す疑似乱数発生器41は、Ex. NORを使用した例である。

【0060】したがって、マイクロプロセッサ等におけるプログラムが解析された場合またはその可能性が生じた場合にバージョンアップして対応するればよく、このバージョンアップ時に、図8に示すLFSR 42の接続状態を変更すればよい。たとえば、バージョン1において、接続点 $c_1$ 、 $c_n$ の2つのみを接続し(Ex. OR $_1$ のみを使用し)、その後、バージョン2において、接続点 $c_1$ 、 $c_2$ に加えて接続点 $c_n$ を接続する(Ex. OR $_1$ とEx. OR $_2$ とを使用する)。バージョン3以降においては、接続すべき接続点の箇所を、上記とは異なる組み合わせにすればよい。

【0061】また、疑似乱数発生器40、41、図8に示すLFSR 42のようなLFSRを基本とする手段とは別の疑似乱数発生器を使用するようにしてもよい。たとえば、2つつ増加する数列を出力する疑似乱数発生器を使用するようにしてもよく、1つつとか2つつとか減少する数列を出力する疑似乱数発生器を使用するよう

してもよい。また、一定の数つつ加算するのではなく、1つつ加算するという法則を除く所定の法則に従った数を、所定の初期値に加算した数で構成される数列を発生する疑似乱数発生器を使用するようにしてもよく、さらには、所定の法則に従った数を、所定の初期値から減算した数で構成される数列を発生する疑似乱数発生器を使用するようにしてもよい。

【0062】つまり、疑似乱数発生手段として、LFSRを基本とする手段と、1以外の所定数つつ増加する数列を出力する手段と、所定数つつ減少する数列を出力する手段と、所定の法則(1つつ加算するという法則を除く法則)に従った数を、所定の初期値に加算した数で構成される数列を出力する手段と、所定の法則に従った数を、所定の初期値から減算した数で構成される数列を出力する手段とのうちのいずれか1つの手段を使用するようにしてもよい。

【0063】上記実施例は、マイクロプロセッサに着目したものであるが、メモリ自体に着目して発明を把握することができる。つまり、1つつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段によって発生される値をアドレスとし、この発生されるアドレスの順序に従って、プログラムが格納されているメモリが本発明であると考えてもよい。

【0064】さらには、メモリ装置に着目して発明を把握することができる。つまり、1つつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段と、上記疑似乱数発生手段が発生する値をアドレスとし、この発生されるアドレスの順序に従って、プログラムが格納されているメモリとを有するメモリ装置が本発明であると考えてもよい。この場合、疑似乱数発生手段は、その初期値を任意に設定可能な手段であつてもよく、その疑似乱数発生手段として、上記のような種々のものを想定することができる。

【0065】また、所定のメモリと、1つつ増加する数列以外の数列である非1インクリメント数列を発生する疑似乱数発生手段が発生する値をアドレスとし、この発生されるアドレスの順序に従って、所定のプログラムを格納する格納手段とによって構成されるメモリ装置が本発明であると考えてもよい。この場合、バスインタフェース15が、上記所定のプログラムを格納する格納手段の例である。

【0066】

【発明の効果】本発明によれば、疑似乱数発生器が発生する乱数をアドレスとしてメモリにプログラムを格納するので、メモリの内容をアドレス順に把握したとしても、正しい順番でプログラムを認識することができず、プログラムを正常に解析することが困難であるという効果を奏する。

【図面の簡単な説明】

【図1】本発明の一実施例であるマイクロプロセッサM

MPU<sub>1</sub> を示すブロック図である。

【図 2】 上記実施例における疑似乱数発生器 40 の構成例と、疑似乱数発生器 40 が出力する値の例を示す図である。

【図 3】 上記実施例であるマイクロプロセッサ MPU<sub>1</sub> の動作を説明する図である。

【図 4】 疑似乱数発生器 40 と同様の疑似乱数発生器 41 の構成例と、疑似乱数発生器 41 の動作を示すタイミングチャートとを示す図である。

【図 5】 本発明の第 2 の実施例であるマイクロプロセッサ MPU<sub>2</sub> を示すブロック図である。

【図 6】 本発明の第 3 の実施例であるマイクロプロセッサ MPU<sub>3</sub> を示すブロック図である。

【図 7】 上記実施例におけるマイクロプログラム制御回路 23 の具体例を示す図である。

【図 8】 上記実施例における疑似乱数発生器 40 を一般化した LFSR (Linear Feedback Shift Register) 4

2 を示す図である。

【図 9】 従来のマイクロプロセッサ MPU の構成例を示す図である。

【図 10】 上記従来例において、メモリ 30 に格納されているプログラムの一例を示す図である。

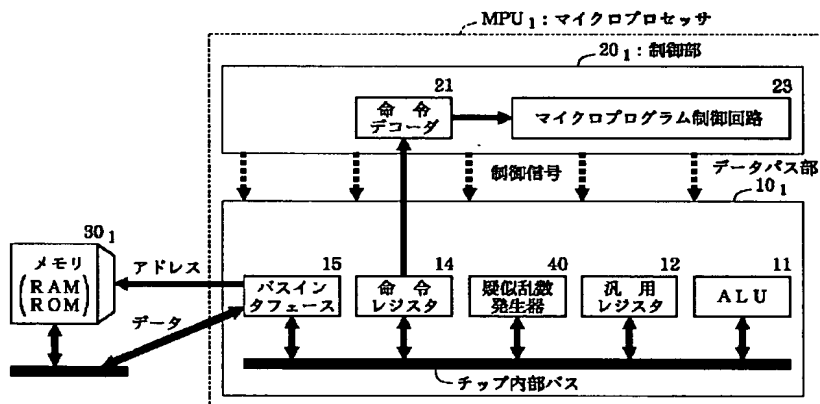
【図 11】 上記従来例におけるプログラムカウンタ 13 の構成の一例を示す図である。

【図 12】 上記従来例におけるマイクロプログラム制御回路 22 の構成例を示すブロック図である。

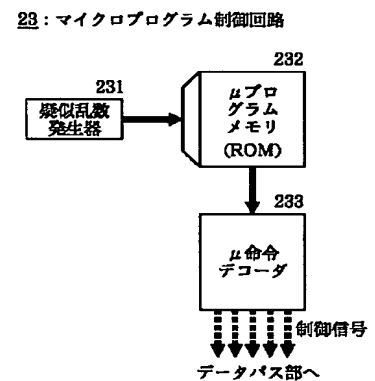
【符号の説明】

MPU<sub>1</sub> ~ MPU<sub>3</sub> ... マイクロプロセッサ、  
10<sub>1</sub> ~ 10<sub>3</sub> ... データバス部、  
20<sub>1</sub> ~ 20<sub>2</sub> ... 制御部、  
30<sub>1</sub> ... メモリ、  
40、41 ... 疑似乱数発生器、  
42 ... LFSR (Linear Feedback Shift Register)。

【図 1】

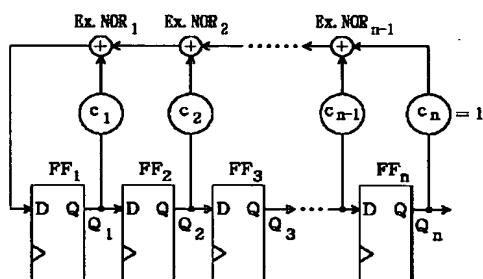


【図 7】



【図 8】

42: LFSR



K4238

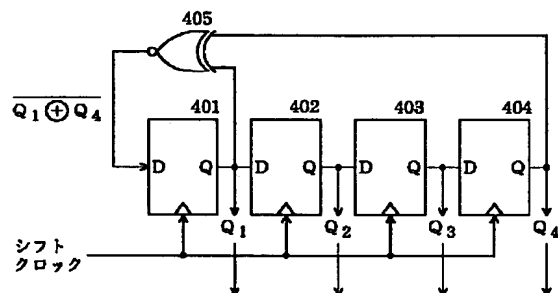
【図 10】

メモリ 30 に格納されているプログラムの例

アドレス	命令コード
0 1 1 1	o p 8
0 1 1 0	o p 7
0 1 0 1	o p 6
0 1 0 0	o p 5
0 0 1 1	o p 4
0 0 1 0	o p 3
0 0 0 1	o p 2
0 0 0 0	o p 1

【図 2】

(1) 40: 疑似乱数発生器



(2)

クロック	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	Q <sub>4</sub>	$\overline{Q_1 \oplus Q_4}$
CL1	0	0	0	0	1
CL2	1	0	0	0	0
CL3	0	1	0	0	1
CL4	1	0	1	0	0
CL5	0	1	0	1	0
CL6	0	0	1	0	1
CL7	1	0	0	1	1
CL8	1	1	0	0	0
CL9	0	1	1	0	1
CL10	1	0	1	1	1
CL11	1	1	0	1	1
CL12	1	1	1	0	0
CL13	0	1	1	1	0
CL14	0	0	1	1	0
CL15	0	0	0	1	0
CL16	0	0	0	1	0

K4236

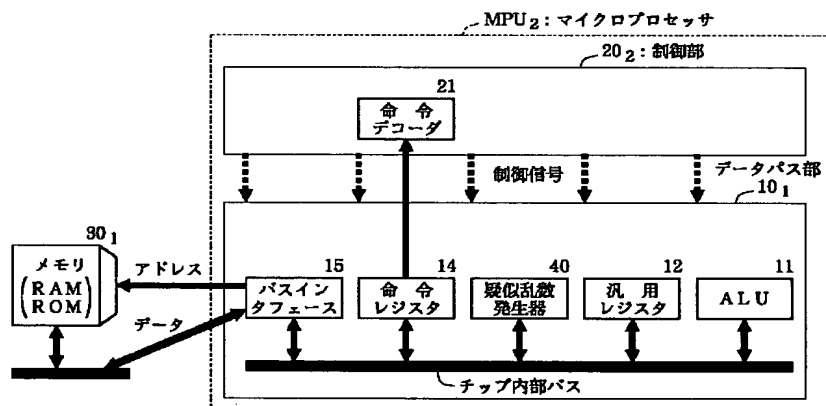
【図 3】

メモリ 30<sub>1</sub> に格納されているプログラムの例

アドレス	命令コード
1110	op12
1101	op11
1100	op8
1011	op10
1010	op4
1001	op7
1000	op2
0111	op13
0110	op9
0101	op5
0100	op3
0011	op14
0010	op6
0001	op15
0000	op1

K4236

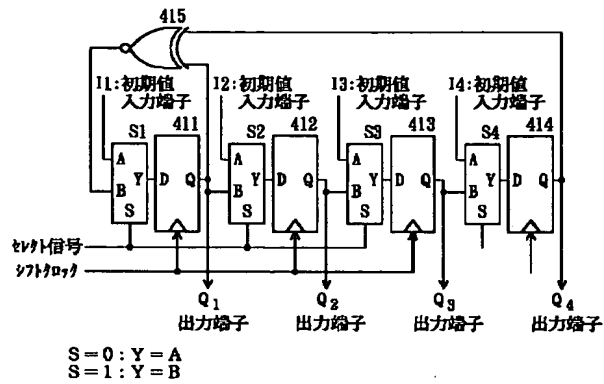
【図 5】



K4236

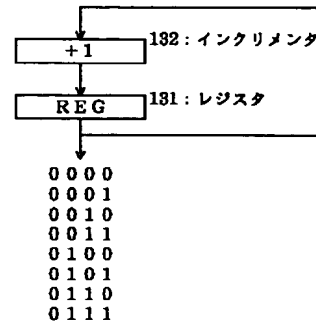
【図 4】

## (1) 41: 疑似乱数発生器



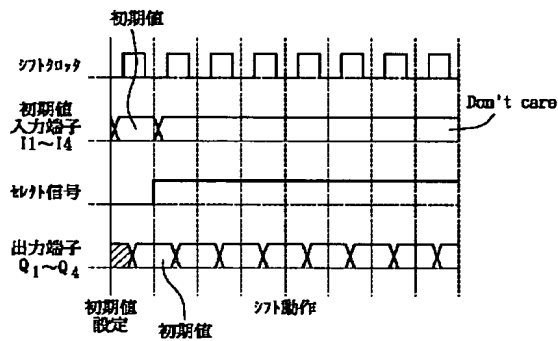
【図 11】

## 19: プログラムカウンタ



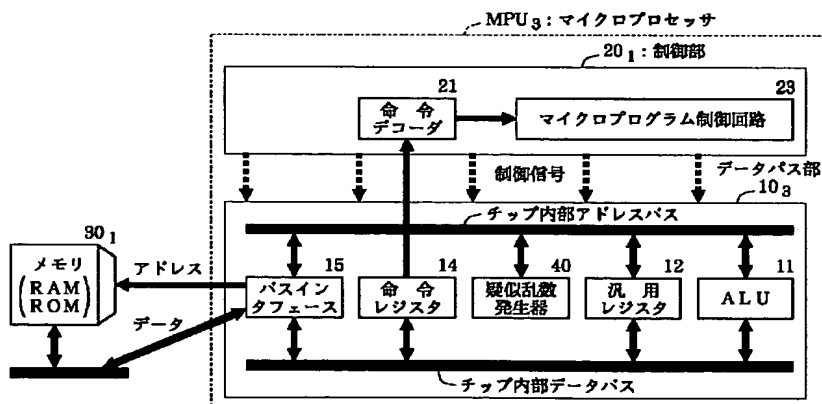
K4236

## (2) 疑似乱数発生器 41 のタイミングチャート



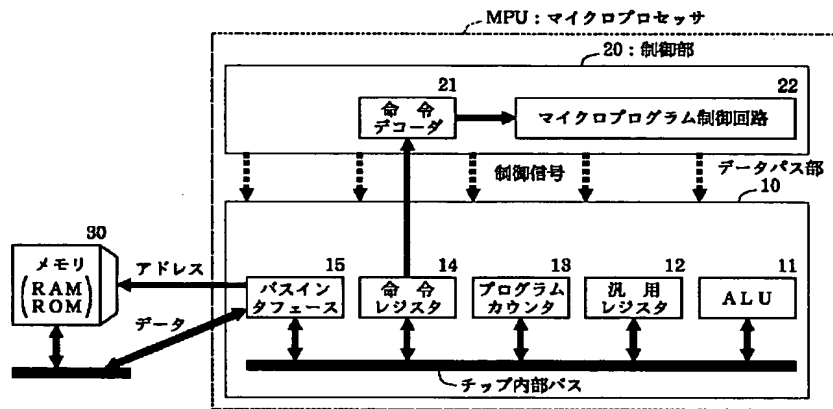
K4236

【図 6】



K4236

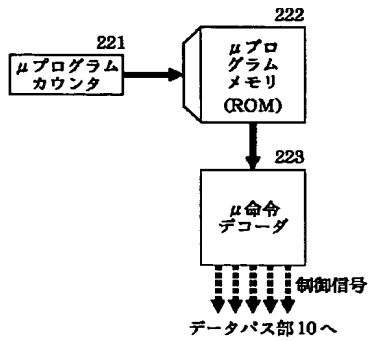
【図 9】



K4237

【図 12】

22: マイクロプログラム制御回路



**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

## **BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☒ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☒ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.